IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of: Hansen, et al. Application No.: 10/757,925 Filed: January 16, 2004 For: METHOD AND SOFTWARE FOR PARTITIONED GROUP ELEMENT SELECTION OPERATION	Examiner: Jesse R. Moll Technology Center/Art Unit: 2181 Confirmation No.: 5116

Mail Stop Amendment
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

DECLARATION OF CRAIG HANSEN UNDER 37 CFR § 1.131

Sir:

I, Craig Hansen, hereby declare the following to be true:

BACKGROUND

1. I am a co-inventor of United States Patent Application No. 10/757,925 (the '925 application). I understand that this application is currently being examined by the patent office. Among other things, this declaration describes certain activities that occurred prior to August 1, 1995, through August 16, 1995, the filing date of U.S. Patent Application Serial number 08/516,036 (the '036 application). The '036 application matured into U.S. Patent number 5,742,840 (the '840 patent), which is the original parent of the '925 application. Attached to this declaration are various documents that evidence conception and diligence in reducing certain inventions claimed in the '925 application to practice.

2. I graduated from Cornell University with a Bachelor's degree in Electrical Engineering. I also received a Master's degree from Stanford University in Electrical Engineering.

3. I began my employment with MicroUnity (assignee of the '925 application) in 1989 and remain employed there today. More specifically, prior to August 1, 1995 and through August 16, 1995, I was employed by MicroUnity as the Chief Architect of the Terpsichore System Architecture, which embodied the inventions of the '925 application. As such, I have personal knowledge of MicroUnity's efforts toward implementing the Terpsichore system.

4. I am currently a shareholder in MicroUnity. I own approximately five percent of the shares of the company on a vested and fully diluted basis, plus approximately one percent additional ownership, when all exercised options become vested. My current position at MicroUnity is Chief Architect.

5. All of the exhibits attached to this declaration are part of MicroUnity's records, and the exhibits are authentic and true copies of original documents (except for the redactions specified below and, in some cases, page numbers used for ease of reference).

CONCEPTION

6. Along with my co-inventor, Dr. John Moussouris, I conceived of a METHOD AND SOFTWARE FOR PARTITIONED GROUP ELEMENT SELECTION OPERATION that includes the following claims:

> 1. A method of processing data in a programmable processor, the method comprising: decoding a single instruction for selectively arranging data, specifying a data selection operand and a first and a second register each having a register width, the first and second registers providing a plurality of data elements each having an elemental width smaller than the register width, the data selection operand comprising a plurality of fields each

independently selecting one of the plurality of data elements; and for each field of the data selection operand, providing the data element selected by the field to a predetermined position in a catenated result.

11. The method of claim 1 wherein for each field of the data selection operand, a relative location of the field within the data selection operand corresponds to a relative location of the predetermined position within the catenated result.

12. The method of claim 1 further comprising: decoding a second single instruction specifying a third and a fourth register each containing a plurality of floating-point operands; multiplying the plurality of floating point operands in the third register by the plurality of operands in the fourth register to produce a plurality of products; and providing the plurality of products to partitioned fields of a result register as a catenated result.

14. A computer-readable medium: having instructions that instruct a computer system to perform operations, at least some of the instructions including a group element selection instruction for selectively arranging data in a programmable processor, the group element selection instruction capable of instructing a computer to perform operations comprising: decoding the group element selection instruction specifying a data selection operand and a first and a second register each having a register width, the first and second registers providing a plurality of data elements each having an elemental width smaller than the register width, the data selection operand comprising a plurality of fields each independently selecting one of the plurality of data elements; and for each field of the data selection operand, providing the data element selected by the field to a predetermined position in a catenated result.

24. The computer-readable medium of claim 14 wherein for each field of the data selection operand, a relative location of the field within the data selection operand corresponds to a relative location of the predetermined position within the catenated result.

25. The computer-readable medium of claim 14 wherein at least some of the instructions further include a group floating point multiply instruction for multiplying floating point data in a programmable processor, the group floating point multiply instruction capable of instructing the computer to perform operations comprising: decoding the group floating point multiply instruction specifying a third and a fourth register each containing a plurality of floating-point operands; multiplying the plurality of floating point operands in the third register by the plurality of operands in the fourth register to produce a plurality of products; and providing the

plurality of products to partitioned fields of a result register as a catenated result.

7. It is my understanding that the claims recited above have been rejected in the outstanding office action based on a patent issued to Ruby Lee and assigned to Hewlett Packard.

8. The conception (before the date of the Ruby Lee patent) of the above-recited claims of the METHOD AND SOFTWARE FOR PARTITIONED GROUP ELEMENT SELECTION OPERATION is shown, in part, in Exhibits 1-2, attached hereto. Certain dates of Exhibits 1-2 showing conception have been redacted in accordance with standard practice and indicated with the phrase "REDACTED". Notwithstanding these redactions in Exhibits 1-2, all of the redacted dates are prior to August 1, 1995.

Exhibits 1 and 2 – Presentation to Cray Research, Inc. and the pre-filing version of the Terpsichore System Architecture Manual

9. Exhibit 1 is a presentation made to Cray Research, Inc. pursuant to a confidentiality agreement between MicroUnity and Cray Research, Inc., with dates redacted. The presentation is a technology overview of the Terpsichore System Architecture and further describes certain subject matter disclosed and claimed in the '925 application. I, along with my colleagues, presented Exhibit 1 to Cray Research, Inc. prior to August 1, 1995. Exhibit 1 contains Bates numbered pages (such as MU0020400) and all references made to Exhibit 1 will refer to the Bates numbered pages.

10. Exhibit 2 is a version of the Terpsichore System Architecture manual that I authored prior to August 1, 1995, with the dates redacted. The manual describes certain subject matter disclosed and claimed in the '925 application. This manual was updated

while work on the Terpsichore system progressed. A later version of this manual, dated August 2, 1995, was filed as an appendix in the '036 application.

11. Exhibit 1 depicts a block diagram of the Euterpe processor at MU0020439. The Euterpe processor is a programmable processor that processes data in accordance with instructions, in a manner that embodies the inventions of the claims. The programmable processor may be used as part of a computer system to perform operations according to programmed instructions. Specifically, the block diagram of Exhibit 1 shows the instruction path of the Euterpe processor, which includes the I Cache/ I Buffer, the instruction queue block, and the issue control block. *Id.* This block diagram also shows data paths connecting the Register Files and Bypass Network block to blocks such as the arithmetic logic unit (ALU), the Galois field (GF), and the local translation look-aside buffer (LTLB) blocks. *Id.* The Register Files are also coupled to the output of the mux (at the bottom of MU0020439 and directly beneath the Register File), making the Register Files capable of receiving data from the data path. The information in this paragraph relates to at least the following portions of the above-cited claims: processing data in a programmable processor; instruct a computer system to perform operations.

12. At MU0020396, Exhibit 1 shows, in the lower half of the page, the G.SELECT.8 instruction. The G.SELECT.8 instruction is a single instruction used for selectively arranging data in a flexible manner. As shown, the bytes within 128 bits of data (sixteen data elements, each having an elemental width of one byte) can be re-arranged under the control of a data selection operand. The data selection operand comprises sixteen fields, each four bits wide. Each field therefore can independently select any one of the sixteen data elements, to be provided to a corresponding position in the result. The instruction

thus performs a 16-way multiplexing of the 128 bits of data with byte-level granularity. In other words, a complete byte permutation on the 128 bits of data can be achieved. As shown in Exhibit 1 at MU0020365, the 128 bits of data would be provided by at least two 64-bit registers in the register file of the Euterpe processor. The register file is also shown in the right portion of the block diagram of the Euterpe processor. *See* Exhibit 1, MU0020439. The information in this paragraph relates to at least the following portions of the above-cited claims: decoding a single (group element selection) instruction for selectively arranging data, specifying a data selection operand and a first and a second register each having a register width, the first and second registers providing a plurality of data elements each having an elemental width smaller than the register width, the data selection operand comprising a plurality of fields each independently selecting one of the plurality of data elements; and for each field of the data selection operand, providing the data element selected by the field to a predetermined position in a catenated result.

13. At MU0020396, Exhibit 1 shows that for the G.SELECT.8 instruction, each of the sixteen fields of the data selection operand selects one of the sixteen data elements of the 128 bit data, to be provided to a corresponding position in the result. The relative position of each field within the data selection operand corresponds to the relative location of the predetermined position in the result where the selected data element is provided. The information in this paragraph relates to at least the following portions of the above-cited claims: for each field of the data selection operand, a relative location of the field within the data selection operand corresponds to a relative location of the predetermined position within the catenated result.

14. At MU0020372, Exhibit 1 shows a listing of instructions performed by the processor, including the "GFMUL" (group floating-point multiply) instruction. The GFMUL instruction multiplies a plurality of floating-point operands in one register by a plurality of floating point operands in another register, to produce a plurality of products, and providing the plurality of products to a result register as a catenated result. At MU0023326-0023328, Exhibit 2 provides further details on specific instances of the GFMUL instruction. These include GF.MUL.16 (group floating-point multiply half), GF.MUL.32 (group floating-point multiply single), and GF.MUL.64 (group floating-point multiply double). As Exhibit 2 shows, each of these instructions takes the contents of registers ra and rb and combines them using the specified floating-point operation (in this case, the operation is multiply). *See* MU0023327, "Description" section. Specifically, the plurality of floating-point operands in register ra is multiplied by the plurality of floating-point operands in rb. *See id.*, "Definition" section. This produces a plurality of floating-point results, which are returned as a catenated result in register rc. *See id.* The information in this paragraph relates to at least the following portions of the above-cited claims: decoding a second single (group floating point multiply) instruction (for multiplying floating point data in a programmable processor) specifying a third and a fourth register each containing a plurality of floating-point operands; multiplying the plurality of floating point operands in the third register by the plurality of operands in the fourth register to produce a plurality of products; and providing the plurality of products to partitioned fields of a result register as a catenated result.

15. As indicated above, Exhibits 1 and 2 were prepared prior to August 1, 1995, prior to which I spent a considerable amount of time conceiving and developing the

Terpsichore System in conjunction with Dr. Moussouris. Based on the aforementioned development effort and as indicated by the accompanying Exhibits, Dr. Moussouris and I conceived the fundamental features of the Terpsichore System prior to August 1, 1995, which we believed would work for their intended purpose once sufficient prototyping and testing efforts were performed.

DILIGENCE

16. From just prior to August 1, 1995 through August 16, 1995, I and others at MicroUnity, as well as MU's legal patent prosecution team, were diligent in our efforts to perform detailed design, to build and to test the Terpsichore system and to file the '036 application, which matured into the '840 patent, which is the original parent of the '925 application. MicroUnity's efforts to further design, build and test the Terpsichore system included the incorporation of features conceived prior to August 1, 1995 into integrated circuitry implementing the Terpsichore system and the development work associated with such incorporation.

17. The evidence showing diligence includes: attorney billing records from MicroUnity's patent prosecution team; modifications made to the electronic databases used to manufacture integrated circuits implementing the Terpsichore system; email communications among the MicroUnity design team making modifications to the electronic databases; and MU payroll records showing salary payments to team members.

Exhibit 3 – Attorney Billing Records

18. Exhibit 3 is an invoice from the law firm of Willian, Brinks, Hofer, Gilson & Lione, MicroUnity's patent prosecution counsel. Descriptions of the services provided to MicroUnity have been redacted to maintain confidentiality/privilege. This invoice,

numbered 78023 (invoice number shown in the upper right hand portion of the page), indicates that Willian, Brinks worked on the patent application (ultimately filed as the '036 application) on August 1, August 8, and August 16 of 1995. This represents part of the substantial amount of work that was performed by MicroUnity's patent prosecution team in completing the patent application. These efforts culminated with the filing of the '036 patent application on August 16, 1995, as indicated by the last entry on the invoice.

Exhibits 4A-4D – Logs of Modifications to the Electronic Databases

19. As discussed above, MicroUnity's Terpsichore system was intended to be implemented in integrated circuit form. *See, e.g.*, Exhibit 1 at pages MU0020321-356 (discussing the status of MicroUnity's integrated circuit fabrication facility). Integrated circuit fabrication facilities, often called "fabs", manufacture integrated circuits on wafers of semiconducting material, which is usually silicon. The location of the circuitry on these wafers, and ultimately the function of the circuitry, is determined by a series of steps that include shining light through reticles to generate the desired circuitry. These reticles are produced using very elaborate electronic databases that typically involve a great deal of time and money to produce. Sometimes, these electronic databases are also referred to as "tapeouts" or "physical layouts" of the integrated circuits in question.

20. The Terpsichore system design team included at least 35 individuals (see Exhibit 6 below). In my capacity of Chief Architect, I was kept abreast of the design team's efforts to implement the Terpsichore system. From a time prior to August 1, 1995, through August 16, 1995, the individuals on this design team spent a substantial amount of their time in building these elaborate databases for the Terpsichore system. Exhibits 4A-4D represent weekly logs of modifications to the electronic databases for the

Terpsichore system from just prior to August 1, 1995, through August 16, 1995. Exhibit 4A is a summary, on a weekly basis, of the number of modifications made to these electronic databases during this time period. In just the few weeks shown by Exhibits 4A-4D, there were approximately 129 changes made to the Terpsichore system electronic databases.

21. Exhibit 4B shows the modification for the week beginning on July 29, 1995. These modifications included approximately 19 changes to the Terpsichore system. (Exhibit 4A). For example, on August 3, 1995, Jeff Marr "[debugged] signals to be used to look for cause of nb shrinkage" (Exhibit 4B at 4). Numerous other modifications were made to the Terpsichore system's electronic database during this week as shown in Exhibit 4B.

22. Exhibit 4C shows the modification for the week beginning on August 5, 1995. These modifications included approximately 71 changes to the Terpsichore system. (Exhibit 4A). For example, on August 8, 1995, Mark Semmelmeyer fixed "[s]ome ICC and CC wires [that] were [unconnected]" (Exhibit 4C at 29) and on August 10, 1995, Jay Tomlinson (referred to in Exhibit 4C as "Woody") added information to the electronic database in order to allow "the [input-output] clock paths [to be tested]" (*Id.* at 26). Numerous other modifications were made to the Terpsichore system's electronic database during this week as shown in Exhibit 4C.

23. Exhibit 4D shows the modification for the week beginning on August 12, 1995. These modifications included approximately 39 changes to the Terpsichore system. (Exhibit 4A). For example, on August 13, 1995, Patricia Mayer "added 3 new resistors" (*Id.* at 7) and on August 16, 1995, Kurt Wampler "[a]dded notes on final Euterpe hand-

routing strategy" (Exhibit 4D at 1). Numerous other modifications were made to the Terpsichore system's electronic database during this week as shown in Exhibit 4D.

Exhibits 5 – Emails among the MicroUnity Design Team

24. Further evidence of MicroUnity's efforts to implement the Terpsichore system in integrated circuit form is shown in email communications among the members of MicroUnity's design team from the time just prior to August 1, 1995, through August 16, 1995. These emails are grouped and attached hereto as Exhibits 5.

25. As can be appreciated from inspection of Exhibit 5, during this time period the design team was continually working to implement the Terpsichore system in integrated circuit form. For example, on Sunday, August 6, 1995, Dave Van't Hof announced that an LVS of the full chip Euterpe had finished (Exhibit 5 at 117). LVS, which stands for layout-versus-schematic, is a tool commonly used to ensure that the electronic databases indeed represent the schematics the designers intended. Also, on Monday, August 7, 1995, a meeting was held where Euterpe was discussed including its ability to complete "between 0.3 and 0.4 instructions/cycle in a single thread." (Exhibit 5 at 108). Further, on Tuesday, August 15, 1995, a review of Euterpe's electronic database or "layout" was conducted where items requiring fixing were identified (Exhibit 5 at 6). Numerous additional implementation details of the Terpsichore system during this time period are evident from inspection of Exhibit 5.

Exhibit 6 – Payroll Records

26. Exhibit 6 reflects various MicroUnity payroll records from just prior to August 1, 2005 through August 16, 1995. The first page is a summary of head count from the three different departments at MicroUnity (Dept. Nos. 310, 320, and 350) performing work on

the Terpsichore system as well as expenditures associated with payroll for these individuals. As can be seen from the summary and accompanying pages, MicroUnity spent approximately $138,000 to $139,000 per pay period from just prior to August 1, 1995 through August 16, 1995 to develop the Terpsichore system.

27. I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true, and the these statements are made with knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code, and that such willful false statements may jeopardize the validity of the application, and any patent issuing thereon, or any patent to which this declaration is directed.

7-Jan-2009
Date

Craig Hansen